Every intellectual cognitive activity is definitely based on Knowledge.

Knowledge is power.

"The power of AI systems resides in the knowledge"
Human beings accumulate billions of 'chunk' of knowledge, connected together and combined in different ways:

- Knowledge about the world
- Knowledge of specific areas
- Knowledge about the human
Everything is intimately linked:

- The basic trio: Knowledge, Reasoning, Memory (where knowledge is stored)
- Overlapping cognitive processes, exploiting the basic trio: Learning, Planning & NLP
- The means of perception (the input / output channels)
At the basic level: use of operations to compare and manipulate knowledge:
- Join two knowledge chunks
- Find knowledge common to two knowledge chunks
- Check if knowledge is contained in another
- etc.
Inferring knowledge from other knowledge. Take into account:

- Types of inference (deduction, induction, abduction, analogy, etc.)

- Degree of certainty (knowledge) of knowledge => approximate reasoning

- The temporal character of all knowledge
The human being is intelligent because it is a 'machine' which consumes and generates continually knowledge.

Important question for AI: how to represent knowledge?

The mode of representation has an impact on any process that manipulates the knowledge.
Procedural representation: compact but difficult to present, extend, exploit, etc.

Declarative representation: Independent description of use, easy to extend and modify
A good system for the representation of complex knowledge structures in a particular domain should possess the properties (Rich 1983):

- **Representational Adequacy**: the ability to represent all the kinds of knowledge that are needed in that domain.
- **Inferential Adequacy**: the ability to manipulate the representational structures in such a way as to derive new structures corresponding to new knowledge inferred from old ones.
- **Acquisitional Efficiency**: the ability to acquire new information easily. The most simplest case involves direct insertion, by a person, of new knowledge in the KB.

What is a good representation?
What is a representation?

- A representation is a set of syntactic and semantic conventions that make it possible to describe things.

- The syntax of a representation specifies the symbols that may be used and the ways those symbols may be arranged.

- The semantics of a representation specifies how meaning is embodied in the symbols and in the symbols arrangements allowed by the syntax.
Formalisms to represent knowledge

- Propositional logic
- Predicate logic
- Frames
- Semantic network
- Conceptual Graph
- etc.
We need a formal notation to represent knowledge

- Allowing automated inference and problem solving

One popular choice is to use logic

Proposition logic is the simplest form of logic

- Symbols represent facts/propositions: p, q, etc.
- We evaluate the truth value of a proposition
- We don’t evaluate the meaning
Propositional logic

- Simple propositions
  - Example: earth is flat

- Composed propositions
  - Example: earth is flat and earth is a planet

- Simple propositions are joined by logical connectives (and, or, negation, implication)
  - \( P \land Q ; \quad P \lor Q ; \quad Q \rightarrow R , \quad \neg S \)

- Given some statements in the logic we can deduce new facts
To derive true formulas from other true formulas, rules of inference are needed.

In a sound theory, the rules of inference preserve truth.

If all formulas in the starting set are true, only true formulas can be inferred from them.

Some of the rules of inference for the propositional calculus are as follows:

- Let symbols p, q and r represent any formula:

**Modus Ponens:** From p and p → q, derive q

**Modus Tollens:** From ¬ q and p → q, derive ¬ p

**Hypothetical Syllogism:** From p → q and q → r, derive p → r

**Disjunctive Syllogism:** From p ∨ q and ¬ p, derive q

**Conjunction:** From p and q, derive p ∧ q
Meaning in propositional logic is context-independent
- unlike natural language, where meaning depends on context

Limits of Propositional logic
- Propositional logic is not powerful enough as a general knowledge representation language
- Impossible to make general statements
- Example:
  - all students take exams
  - if any student take an exam, s/he either passes or fails
Demonstrate that $p \rightarrow q$ is equivalent to $\neg(p \land \neg q)$

- We have the succession of equivalences
- $p \rightarrow q \iff \neg p \lor q$ (implication elimination)
- $\neg p \lor q \iff \neg p \lor \neg \neg q$
- $\neg p \lor \neg \neg q \iff \neg(p \land \neg q)$ (De Morgan)
Formalisms to represent knowledge

- Propositional logic
- Predicate logic
- Frames
- Semantic network
- Conceptual Graph
- etc.
Whereas propositional logic assumes the world contains facts, first-order logic (like natural language) assumes the world contains:
- Objects: people, houses, numbers, colors, wars, etc.
- Relations: red, round, prime, brother of, part of, etc.

In predicate logic the basic unit is a predicate/argument structure called an atomic sentence:
- 
  \[ \text{likes}(\text{ali}, \text{chocolate}) \]
  \[ \text{tall}(\text{zakaria}) \]

Arguments can be any of:
- constant symbol, such as ‘\text{ali}’
- variable symbol, such as \text{x}
- A function, such as \text{sqrt}(\text{n})

Examples:
- 
  \[ \text{Likes}(\text{X}, \text{chocolate}) \]
  \[ \text{Friends}(\text{zakaria}, \text{youssef}) \]
These atomic sentences can be combined using logic connectives:
- \( \text{likes(ali, chocolate)} \land \text{tall(zakaria)} \)
- \( \text{tall(zakaria)} \rightarrow \text{play(zakaria, basket-ball)} \)

Sentences can also be formed using quantifiers \( \forall \) (for all) and \( \exists \) (there exists) to indicate how to treat variables:
- \( \forall X (\text{mortal}(X)) \) Everything is mortal
- \( \exists X (\text{mortal}(X)) \) Something is mortal
- \( \forall X (\text{on}(X, \text{earth}) \rightarrow \text{mortal}(X)) \) Everything on earth is mortal

We can have several quantifiers in an expression, such as:
- \( \forall X \exists Y (\text{father}(X, Y)) \)
- \( \forall X (\text{expensive}(X) \rightarrow \exists Y (\text{wants}(Y, X))) \)

Here are identities common in predicate calculus:
- \( \exists X (P(X)) \) is identical to \( \neg \forall X (\neg P(X)) \)
- \( \forall X (P(X)) \) is identical to \( \neg \exists X (\neg P(X)) \)
We can define inference rules allowing us to say that if certain things are true, certain other things are sure to be true, e.g.

\[ \forall X (P(X) \rightarrow Q(X)) \]

\[ P(aa) \]

----------------------- (so we can conclude)

\[ Q(aa) \]

This involves matching \( P(X) \) against \( P(aa) \) and binding the variable \( X \) to the symbol \( aa \)

\[ \forall x \quad (\text{Chinois}(x) \rightarrow \text{Pere}(Mao, x)) \land \text{Chinois}(Ching) \rightarrow \text{Pere}(Mao, Ching) \]

Example: What can we conclude from the following?

\[ \forall X \quad \text{Tall}(X) \rightarrow \text{Strong}(X) \]

\[ \text{Tall}(john) \]

\[ \forall X \quad \text{Strong}(X) \rightarrow \text{play}(X, \text{Boxe}) \]
Predicate Logic - Exercises

- Represent in terms of predicates:
  - Ahmed gives Ali a book
  - Somebody gives a book to Ali
  - Jacques envoie un livre à Marie
  - Chaque homme se promène
  - Certains hommes se promènent
  - Aucun homme ne se promène
  - Jacques envoie quelque chose à chacun

  - gives (Ahmed, Ali, book)
  - or $\exists x (\text{gives}(\text{Ahmed}, \text{Ali}, x) \land \text{Book}(x))$
  - $\exists y \exists x (\text{Gives}(y, \text{Ali}, x) \land \text{Person}(y) \land \text{Book}(x))$
  - $\text{Envoi}(\text{jacque1}, \text{Marie4}, \text{Livre2})$
  - $\forall x (\text{Homme}(x) \rightarrow \text{Promener}(x))$
  - $\exists x (\text{Homme}(x) \rightarrow \text{Promener}(x))$
  - $\neg (\exists x (\text{Homme}(x) \rightarrow \text{Promener}(x)))$
  - $\exists y \forall x (\text{Envoi}(\text{jacque1}, x, y))$
Propose a definition for \( \text{GrandParentOf}(x, y) \)

Propose a definition for \( \text{Ancestor}(x, y, n) \)

using \( \text{Person}(X), \, \text{ParentOf}(X, Y) \)

\[
\text{GrandParentOf}(x, y) \leftarrow \\
\text{Person}(x) \land \text{Person}(y) \land \\
\exists z \, ( \text{Person}(z) \land \text{ParentOf}(x, z) \land \text{ParentOf}(z, y) )
\]

\[
\text{Ancestor}(x, y, n) \leftarrow \\
\text{Person}(x) \land \text{Person}(y) \land \\
\exists z \,( \text{Person}(z) \land \text{ParentOf}(z, y) \land \text{Ancestor}(x, z, n-1) )
\]

\[
\text{Ancestor}(x, y, 1) \leftarrow \\
\text{Person}(x) \land \text{Person}(y) \land \text{ParentOf}(x, y)
\]
Formalisms to represent knowledge

- Propositional logic
- Predicate logic
- Frames
- Semantic network
- Conceptual Graph
- etc.
Frames

- Family of object-oriented languages

- Advantages of object-oriented languages:
  - Data abstraction
  - Modularity / Modifiability
  - Reusability
  - Readability / Understanding
  - Heritage

- Language of frames: OO + procedural attachment

- Frames are prototypes for specifying K that are poorly described in predicate calculus: typicality, default values, incomplete information.
In the language of frames:
- An object = unit of K representing the prototype of a concept
- A frame is a generic entity composed of attributes (slots) that describe the different properties of the represented concept

The set of frames is organized according to a hierarchy where each object is both a representation of the frames from which it is derived and a generator of frames more specialized

- A frame has attributes whose various aspects are described by declarative facets (trigger)

- A frame does not have its own behavior described by methods
Définition: x|DAY

- `year`: (y|INTEGER)
- `month`: (`when-filled` (check MONTH))
- `day`: (`when-filled` (check-day))
- `day-of-week`: (`to-fill` (get-day-of-week))

Instance: DAY 124

- `self`: (ELEMENT-OF DAY)
- `year`: 1981
- `month`: 8
- `day`: 3
- `day of week`: MONDAY
Formalisms to represent knowledge

- Propositional logic
- Predicate logic
- Frames
- Semantic network
- Conceptual Graph
- etc.
Exploit the connectivity of a graph to represent the connectivity between the concepts:

- Often, a situation is described by a conjunction of predicates that share common arguments
- Connectivity is implicit in the predicate formulation

SN: set of concepts connected by relations
Semantic networks attempt to combine in a single mechanism the ability to store factual information and to model the associative connections between information items exhibited by humans.

Network designation corresponds to the method where the concepts are represented by nodes, and relations (in general binary) by labeled arcs.
Inheritance via isa link
Phrase 1 : Jacques écrit un livre, Phrase 2 : Jacques envoie ce livre à Marie
Phrase 3 : Marie lit le livre.

Sens et Contexte
John hits Mary with a flower

- There is a problem with this representation: There are 3 instances defined in the ‘fact base’: john, mary, flower33

- But what is ‘hits’? ‘hits’ should be replaced by hit46, an instance of the generic action HIT (HIT is an action type)

- If we explicitly introduce all the needed types, what is the new semantic network representing the sentence ‘John hits Mary with a flower’?
Semantic network Example
Formalisms to represent knowledge

- Propositional logic
- Predicate logic
- Frames
- Semantic network
- Conceptual Graph
- etc.
A formal language developed by John Sowa (IBM)

Based on Semantic Networks

Human readable

CGs express logical precise meaning

Suitable for representing natural language
Constituents of CGs

- Two kinds of nodes
  - Concepts – boxes / [square brackets]

[Cat] [Mat]
Conceptual Graphs

Constituents of CGs

- Two kinds of nodes
  - Concepts – boxes / [square brackets]
  - Conceptual relations – circles / (parentheses)

[Cat] (on) [Mat]
Constituents of CGs

- Two kinds of nodes
  - Concepts – boxes / [square brackets]
  - Conceptual relations – circles / (parentheses)

- Arches between concepts and relations
  - CGs are bipartite graphs

[Cat] \(\rightarrow\) (on) \(\rightarrow\) [Mat]
Display form and linear form

[Cat] \(\rightarrow\) (on) \(\rightarrow\) [Mat]

[graph:{*}] \(\rightarrow\) (attr) \(\rightarrow\) [conceptual]
There exists a concept: ‘Cat’
Constituents of CGs

There exists a concept: ‘Cat’
Constituents of CGs

[Cat] \rightarrow (on) \rightarrow [Mat]

- There exists a concept: ‘Cat’ and a concept ‘Mat’
Constituents of CGs

There exists a concept: ‘Cat’ and a concept ‘Mat’*
And the cat is on the mat
Constituents of CGs

\[[\text{Cat}] \leftarrow (\text{on}) \leftrightarrow [\text{Mat}]\]

- There exists a concept: ‘Cat’ and a concept ‘Mat’*
- And the mat is on the cat
- Reading graphs

[concept] → (relation) → [concept]

[concept] ← (relation) ← [concept]
Reading graphs

[concept] → (relation) → [concept]

has a which is

[concept] ← (relation) ← [concept]

is a of
Reading graphs

[concept] \rightarrow (relation) \rightarrow [concept]

has a which is

[concept] \leftarrow (relation) \leftarrow [concept]

is a of

[walk] \rightarrow (agnt) \rightarrow [person:john]

walk has an agent which is john

[person:john] \leftarrow (agnt) \leftarrow [walk]

john is the agent of walk
Reading graphs

[chase] -
  -(agnt) \rightarrow [dog]
  -(ptnt) \rightarrow [cat]

[chase] -
  -(ptnt) \rightarrow [dog]
  -(agnt) \rightarrow [cat]
Conceptual Graphs

- Concept: [Type: Referent = Descr]

![Conceptual Graph Example](image-url)
Concept : [Type : Referent = Descr]

In graphical notation
notation CGIF

(agt r Fatima) (obj r ProjA) (agt Fatima s)
(obj s [Service]) (chrc ProjA [Complex])
(attr Fatima [kind])
[Girl= Yasmine]←agnt-[love]-obj→[Dance]

\[\exists x, \exists y \ ( \text{Girl}(Yasmine) \land love(x) \land Dance(y) \land agnt(x,Yasmine) \land obj(x,y))\]
Ahmed thinks it is possible to write a good Assembling program

\[
\begin{align*}
\text{[Person:Ahmed]} & \leftarrow \text{agt} - \text{[think]} - \text{obj} \rightarrow \text{[Proposition]} = \\
& \leftarrow \text{attr} - \text{[Proposition]} = \\
& \leftarrow \text{[Person]} \leftarrow \text{agt} - \text{[write]} - \text{obj} \rightarrow \text{[Program]} - \\
& \text{-attr} \rightarrow \text{[good]} \\
& \text{-chrc} \rightarrow \text{[Assembling]}
\end{align*}
\]
Composed GCs

Man : Hicham

agent

 objeto

prop

queen

Woman : Karima

Vase

Color: white

Attr

comp

glass

dirham: 750

prop

Vase

manufactured

lieu

China

Table

circular

form

Sur

Vase

Color: white

Comp

Prox

crystal

obj

believe

agent

fixe

Pat

Sit-down

Obj

Believe

obj
Represent the sentence: “Ahmed gives Ali a book”
- using the graphical form
- using the linear form
- in Predicate calculus

Represent using the graphical form of CGs the sentences:
- Mary buys a flower and John eats an apple
- Ahmed believes Ali is walking in the park

Represent using the graphical form of CGs the sentences:
- Mary buys an apple and John eats it
- Jean believes that Mary will give him a flower
Represent this sentence using the linear form of CGs
[HIT] –

(Agnt) -> [PERSON: John]
(Recip) -> [PERSON: Mary]
(Inst) -> [FLOWER: *]

Represent this sentence in Predicate calculus
\[ \exists x \exists y \ ( \text{Person}(John) \land \text{Person}(Mary) \land \text{Flower}(x) \land \text{Hit}(y) \land \text{Agnt}(y, John) \land \text{Recep}(y, Mary) \land \text{Inst}(y,x) ) \]
SITUATION:

Apple: *

Obj → Buy → Agnt → PERSON: Marie

T: *x

Obj → Eat → Agnt → PERSON: Jean

PROPOSITION:

PERSON: Jean *x

Agnt → Believe → Thm

Flower: *

OBJ → Give → Agnt → PERSON: Mary
Valence

- The valence of a relation is the number of arcs that belongs to it.

- Monadic relation: a relation to which one arc belongs

- Dyadic relation

- Triadic relation
  (office_2 is between office_1 and office_3)
[event= \[act: \text{hit}\] -
  -(agnt) \rightarrow \text{[person: John]}
  -(ptnt) \rightarrow \text{[object: vase }^\times\text{]}
  -(inst) \rightarrow \text{[artefact: bat]}
  -(rslt) \rightarrow
  [state= \text{[object: }^?x\text{]} \leftarrow \text{(thme)} \leftarrow \text{[adjective: broken]}\]
Thematic roles proposed by Sowa (2000)

<table>
<thead>
<tr>
<th>Action</th>
<th>Source</th>
<th>Product</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Initiator</td>
<td>Resource</td>
</tr>
<tr>
<td></td>
<td>Agent, Effector</td>
<td>Instrument</td>
</tr>
<tr>
<td>Process</td>
<td>Agent, Origin</td>
<td>Matter</td>
</tr>
<tr>
<td>Transfer</td>
<td>Agent, Origin</td>
<td>Instrument, Matter</td>
</tr>
<tr>
<td>Spatial</td>
<td>Origin</td>
<td>Path</td>
</tr>
<tr>
<td>Temporal</td>
<td>Start</td>
<td>Duration</td>
</tr>
<tr>
<td>Ambient</td>
<td>Origin</td>
<td>Instrument, Matter</td>
</tr>
</tbody>
</table>
Final example
Ontological description of types (concepts) and relations

Organize concepts and relations in a hierarchy (graph)
Definition of an ontology (wikipedia)

Ontology is the philosophical study of the nature of being, existence or reality in general, as well as of the basic categories of being and their relations. ... what entities exist or can be said to exist, and how such entities can be grouped, related within a hierarchy, and subdivided according to similarities and differences.

Definition of an ontology (Sowa)

The subject of ontology is the study of the categories of things that exist in some domain. The product of this study, called an ontology, is a catalog of the types of things that are assumed to exist in a domain of interest D from the perspective of a person who uses a language L for the purpose of talking about D.
- Defining elements
  - Defining concepts
  - Defining relations
Concepts

- A concept consists of a type and a referent
Concepts

- A concept consists of a type and a referent
- **Concepts**
  - A concept consists of a type and a referent
  - A concept type is a class belonging to a hierarchy
  - A referent is an individual belonging to a class
Small ontology
- Step by step building
Small ontology
- Step by step building
- Small ontology
  - Step by step building
- Small ontology
- Step by step building
- Small ontology
  - Step by step building
- Small ontology
  - Step by step building
Small ontology
- Step by step building
Small ontology

- Universal type
  - Object
    - Inanimate
      - Food
      - Artefact
      - Place
    - Animate
      - Plant
      - Animal
      - Person
  - Process
    - Proposition
    - State
    - Event
    - Act
Small ontology

Ontology
Concept types and individuals

universal >
  object, process, proposition.
object > animate, inanimate.
animate > animal, person, plant.
inanimate > artefact, food, place.
process > event, state.
event > act.
Concept types and individuals

act = buy, chase, sit, talk.  
animal = cat, dog, horse.  
artefact = car, mat, table.  
person = John, Peter, Susan.

universal >
  object, process, proposition.
  object > animate, inanimate.
  animate > animal, person, plant.
  inanimate > artefact, food, place.
  prosess > event, state.
  event > act.
Concept types and individuals

- act = buy, chase, sit, talk.
- animal = cat, dog, horse.
- artefact = car, mat, table.
- person = John, Peter, Susan.

universal >
  object, process, proposition.
  object > animate, inanimate.
  animate > animal, person, plant.
  inanimate > artefact, food, place.
  prosess > event, state.
  event > act.

[act: talk] -
  -(agnt) → [person: John]
  -(rcpt) → [person: Peter]
  -(thme) → [animal: horse]
Concept types and individuals

[act: talk]-
-(agnt)→[person: John]
-(rcpt)→[person: Peter]
-(thme)→[proposition

Embedded graph

[act: buy]-
-(agnt)→[person: Susan]
-(thme)→[animal: horse]

'John tells Peter that Susan buys a horse'
Building ontologies

- bottom-up
- Top-down
Ontology by Sowa

(Sowa 2001a)
Some operations on concepts

- **subType**(Type1, Type2)
  - subType(Man, Person)

- **maxComSubType**(Type1, Type2, Type3)
  - maxComSubType(Man, Person, Man)
  - maxComSubType(Person, Vehicle, null)

- **minComSuperType**(Type1, Type2, Type3)
  - minComSuperType(Man, Person, Person)
  - minComSuperType(Animal, Boy, Living body)
Definition of a concept type: Type art-Sponsor(x) is:

\[ \text{[Person : } x \text{]} \leftarrow \text{agnt-}[\text{Give}]\rightarrow \text{-obj} \rightarrow \text{[Money]} \rightarrow \text{-rcpt} \rightarrow \text{[Artist]} \]

Definition of relation type: Relation brother(x, y) is:

\[ \text{[Boy: } x \text{]} \leftarrow \text{sonOf-}[\text{Person}] \text{-sonOf} \rightarrow \text{[Person : } y \text{]} \]
Canon (the canonical graph providing basic semantics about some type):

Canon for teaching:

\[
\text{[teacher]} \leftarrow \text{agnt-} \text{[teaching]} -
\text{-obj}\rightarrow \text{[Course]},
\text{-rcpt}\rightarrow \text{[Person]}
\]

Canon for Arriving:

\[
\text{[Mobile-entity]} \leftarrow \text{agnt-} \text{[Arrive]} - \text{loc}\rightarrow \text{[Place]}
\]
The Joint of two CG → a CG that groups information contained in both

G1 : [Person]←agnt-[drive]-obj→[car]

G2 : [boy: Hicham]←agnt-[drive]-manr→[fast]

Resultat : [boy: Hicham]←agnt-[drive]-
-obj→[car],
-manr→[fast]
Operations on CGs
contraction

To Contract a graph (habituellement la définition d’un type) of another one:

[Rich]←attr-[man: Hicham]←agnt-[give]-
-obj→[money]
-rcpt→[Artist]

Example of art-Sponsor:

[art-Sponsor: Hicham]-attr→[Rich]
Generalizing two graphs => find the CG that represents common information

G3: [BOY: Hicham] ← agnt-[DRIVE]-obj → [CAR]-chrc→[Color = RED]

G4: [GIRL: Mary] ← agnt-[DRIVE]-

-obj→[VEHICLE],

-manr→[FAST]

Result: [Human] ← agnt-[DRIVE]-obj→[Vehicle]
Amine Platform: CG and Ontology

- AminePlatform